Enhancing Web Supported Learning by Adding a Management Layer to LMSs

Carlos Sousa Pinto
Universidade do Minho, Portugal
csp@dsi.uminho.pt,
Fernando M.S. Ramos
Universidade de Aveiro, Portugal
fmr@ca.ua.pt

Abstract

The success of Web based learning experiences can be compromised by many reasons. For example, if someone does not execute a programmed activity, inside the defined timing, it can compromise the rest of the course to that person. It would be important to detect these situations and to take any corrective action.
It seems to be necessary to use mechanisms of management in real time of the envolvement of each participant in a distance learning course using LMS (Learning Management System), in order to allow the detection of deviations to the scheduled activities for each actor, enabling the correction of these deviations [1], [2].
The standardization works being developed by organizations and consortiums like IMS, SCORM, IEEE, AICC, ARIADNE, etc, don't cover the course monitorization concerns mentioned. Those projects were focused on aspects like contents and its delivery in the context of the actors participation on the courses [3], [4], [5].
This article describes a proposal of reference model and functionalities towards a specification of a layer for real-time management of user interactions on LMSs, and its possible integration with the ADL SCORM standard proposal. The paper includes a discussion of the management metadata model for the LMS sub-system and how the integration of the management module under SCORM may be achieved.

1. Introduction

Thinking on Web based courses, we can identify activities to be executed not only by learners but also by teachers and elements of support teams. The success of those teaching/learning experiences depends on the right participation of each of those types of actors.

Sometimes, for a learner or for all the participants in a course, the objectives established for that course are not reached, and we can identify several reasons for that. For example, if someone does not execute a programmed activity, inside the defined timing, it can compromise the rest of the course to that person.

Let's see another example involving the teacher and the learners. Let us assume that an activity A1 is programmed to be executed by the learners and that it depends on the previous knowledge of the result of the evaluation of a work submitted by the learners to the teacher (activity A2). If the teacher doesn't inform the learners about that classification in useful time, that can compromise the execution of the activity A1. In such situations it would be important to detect the fault and to take some corrective actions. Unfortunately, many times, the identification of the problems that leads to the insuccess occurs too late, compromising any type of solution for those problems.

We belive that LMSs should include mechanisms for automatic monitorization of the participations, so that the probability of success of the teaching/learning process could be enhanced.

2. Our conceptual model

Our proposal for the management layer lies in the monitorization of an informational entity that we call "events" and in its comparison with another one that we assign as "activities". This last one implements the structure of the course while the first reflects the interactions of the actors with the LMS, in what concerns the execution of the scheduled activities.

The subsystem of management completes itself with the inclusion of a component of notifications and with the definition of a set of rules that regulate the notification process.

This functionality foresees the existence of three different instants where notifications can occur:

- Before the beginning of the activity (Warning);
- Before being reached the limit defined for the execution of the activity (First alarm);

- After this limit has been exceeded (Second alarm).

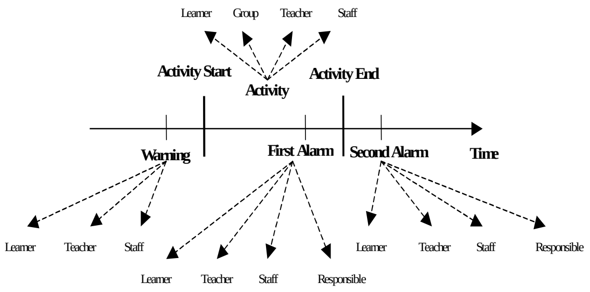

Figure 1. Atomic Unit of Management in Real Time of Activities

Figure 1 represents what we call "Atomic Unit of Management in Real Time of Activities", on the basis of which all the courses can be architected.

For us, a course can be any combination of units of this type, organized in a sequential, parallel or random way and including the possibility of recursive application of this concept to the decomposition of an activity in sub-activities, to be executed by an actor or a group of actors.

As we can see in the Figure 1, our model includes several types of possible actores. In the documentation about the principal projects on this subject [3], [4], [5] only learners are referred and we can't read anything about the participation of groups, teachers and members of support teams.

In the same way, we can't see any references to group activities. That is, activities composed by sub-activities as showed in Figure 2, each of them to be executed by an element of the group.

Figure 2 shows an activity composed by sub-activities, each of them having exactly the same set of proprieties referred above. In that figure we represent the sub-activities as sequencial but it was possible to include sub-activities to be executed in a parallel way.

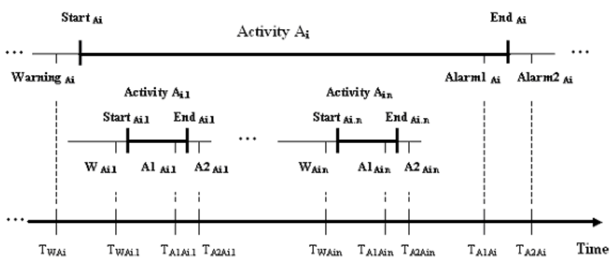

Figure 2. Activity composed by sub-activities

In accordance with Figure 1, each activity has a "warning" emitted before the instant defined for the beginning of the activity, to alert the actors to the proximity of the beginning of that activity. This type of notification makes sense only if the activity is not a random one. In these cases the activity is initiated by the choice of the actor and not by the occurrence of a defined trigger.

When the activity is initiated, its conclusion must occur inside the defined window of time.

Before reaching the deadline to the execution of the activity it must be tested if the activity was already terminated or if it is still running. If this is not the case, a "first alarm" will be generated.

In this way it can be prevented that the structure of the course has to be redefined and the system will potentially contribute for the increase of the probability of success of actors' participation in the course.

Finally, once it is possible that an actor misses the execution of an activity inside the foreseen window of time, the system will have to emit a third type of notification, a "second alarm", destined to make possible the adoption of corrective actions, namely the reprogramming of the activity for this actor.

Figure 1 shows that a course can include activities to be executed by learners, groups, teachers and elements of the team of support.

We can also see that notificatios can be sent to different destinations, depending on the type of notification (warning, first alarm or second alarm).

3. Integrating our work into ADL SCORM

Once there are several organizations and consortiuns, involving the industry, governmental institutions and the universities developing works of standardization, it seemed important to see how the referred management aspects were covered by these works, and to perceive how it could be possible to articulate our work with the ones available from these organizations and consortiuns.

Given the existence of the already referred works of standardization (IMS, AICC, ARIADNE, ADL, IEEE) and once the project ADL SCORM (Sharable Content Object Reference Model) is the one that congregates greater number of contributions from other projects, we thought that it would be interesting to develop our work towards its possible integration in the SCORM. Being so, we made the identification of potential points of interface between our management layer and other layers referred in the SCORM 2004 specification, to allow the monitorization of the interactions with the LMSs. This work leads to the identification of SCORM behaviors and elements of metadata that need to be enhanced and to the definition and inclusion of procedures in our subsystem of management, capable to make compatible this new layer with the functionalities that already exist in the SCORM project.

Figure 3 represents our perspective of the integration of the proposed management layer and the different modules of an LMS and the way they should related to each other.

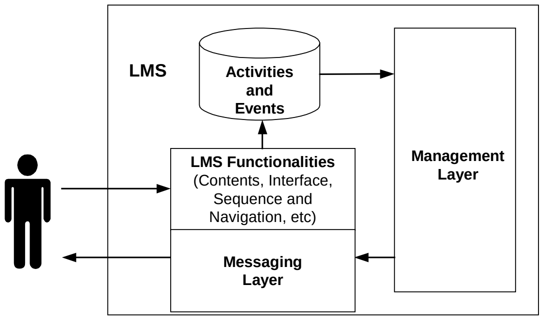

Figure 3. Architecture of management layer relationship with other LMS components

We should read the scheme of Figure 2 as follows:

1. The authors of the courses interact with the platform in order to construct the courses, registering among other information, the one that implements the structure of the course itself, that is, the activities.
2. Later, the actors, in order to execute their activities, will interact with the LMS and, during this interaction, the LMS promotes the register of the diverse corresponding “events".
3. The actors will be able to use the mechanisms of synchronous and/or assynchronous communication, to communicate informally between them.
4. Permanently, the layer of management of the LMS will consult the repository of activities and events to identify situations that justify the emission of notifications. If there are cases that justify this emission, the management layer will request the "messaging" layer of the LMS, passing to it, pairs with the following constitution:

    - Identification of the destination;
    - Message

5. Finally, the LMS using its functionalities of "messaging", after identifying the preferential way of communication of each destination, will send the notifications, according to the information received from the management layer, or it will create the conditions so that these notifications are sent in a non electronic form.

It should be highlighted that we can have more than one destination for a notification, namely when sending messages for a group of learners, for example. Even the case of destinations of different types, eventually receiving different messages, is well supported by our management layer as it can be inferred from the structure of informational pairs showed above in point 4.

In order to articulate our proposed management layer with the LMSs builded under ADL SCORM recomendations, it is necessary that the LMSs can create the information about the execution of the activities in our informational entity “events”. The registration of that information must be done only if the activities are terminated successfuly. In our point of view, an activity for which there is no “event” registration, is an activity not executed and the management layer must generate notifications related to that fact.

We propose that the alterations to ADL SCORM specification should be done in the RTE (Run-Time Environment) documentation, in what concerns the use of the API (Application Programming Interface).

During the execution of an SCO (Sharable Content Object), that was launched by the LMS, it finds the API and iniciates the communication between itself and the LMS by calling the methods pertained to the API. Those methods are distributed by three main groups – Session Methods, Data-transfer Methods and Support Methods.

The session methods – “Initialize()” and “Terminate()” - are used to initiate and terminate the communication while the data-transfer methods – “GetValue()”, “SetValue()” and “Commit()” – are used to manage the storage and retrieval of data to be used in the actual communication session [3]. The method “SetValue()” is used to send information from SCO to LMS, for storage.

We think that it is possible to extend the behavior of this component of the LMS API so that it could promote the insertion of right information in our “events” informational entity.

4. The meta-data model

Figure 4 is the hierarchic meta-data model corresponding to our vision of what a course should be. In that model we represent more than the elements strictly related to the problem of management we are discussing in this paper. In fact, the model represented in Figure 4 is a data model that could support a complete LMS, accordingly to our perspective of what an LMS should be.

It must be highlighted that for text length limits, some elements are not expanded in the Figure 4 but all of them are defined using other elements and atributtes.

The symbols before the elements and attributes have the following meaning:

“+” - The element can exist one or more times.
“*” - The element can exist zero or more times.
“?” - The element is opcional.
“D” - The attribute has a default value.

All the elements and attributes without any precedent symbol are mandatory and must exist one time.

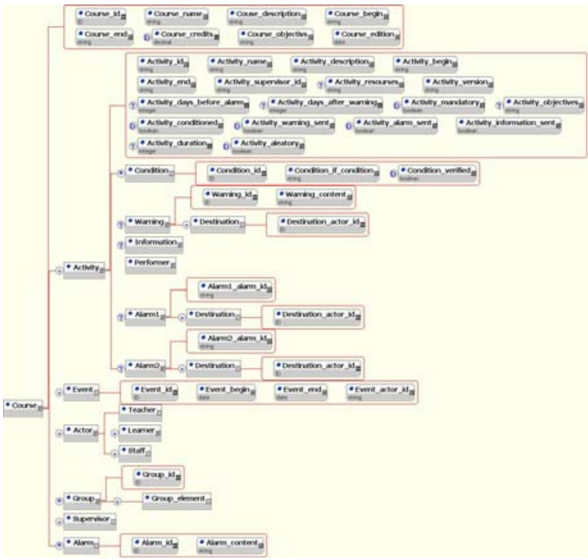

Figure 4. Parcial view of hierarchy of meta-data elements

In the model of the Figure 4 we have included the elements “alarm1” and “alarm2” without a “content” attribute because there are no conceptual differences between the two types of alarms. Only the timing of eventual appearance in the process is different. This way, the two elements have an attribute (Alarm1_alarm_id and Alarm2_alarm_id) that points to the element “alarm” where all the possible alarms must be stored.

It is clear in the meta-data model that an alarm (first alarm or second alarm) can have more than one destination as referred above.

5. Conclusions

The standardization works being developed are very important because they will allow the uniformization of the LMSs and contents development. This is a key aspect in order to obtain greater levels of reuse and interoperability among different systems. However, it is clear that those works have as principal concerns, the contents development, the schedulling of the activities to be executed inside the courses and mechanisms for sequencing and navigation over the contents. Aspects that we consider important, like real-time monitorization of the different actors participation, are not considered.

Our experience in Web based distance learning indicates that when there are not an efective follow up of the activities by the responsible for the courses, the probability of insuccess grows up.

On the other hand, it seems to be an incomplete approach to consider only learners as actors participating in a course and that is what we can see in the different projects documentation, namely in the SCORM and IMS documentation.

Teachers and members of the support teams are also important actors to be considered in courses operationalization and it is very easy to identify several activities to be executed by them.

Based on these considerations we have developed the work presented in this paper, having in mind the proposal of a reference model and functionalities towards a specification of a layer for real-time management of user interactions on LMSs, and its possible integration with the ADL SCORM standard.

Our proposed management layer can detect deviations to the course scheduled activities, enabling the correction of these deviations in useful time. This is possible due to a component of automatic notifications that is also responsible for the detection of abnormal situations.

The validation of the work is not complete at this time. It is necessary to effectively integrate our management layer in a SCORM complient LMS and to use this e-learning platform in a significant number of experiences of distance learning. After these experiences it will be possible to compare the results with those known from passed experiences.

6. References

[1] Ramos, F. Moreira and Santos, A., “Towards a Reference Framework for eLearning Management”, Proceedings of DLA’2001 - Distance Learning Administration Conference, University of West Georgia, GA, USA, (2001), June 6-8.

[2] Ramos, F., Moreira, A., Santos, A., Conde, A., Neves, L., Peixinho, F., Pinto, C. S., “Tracing Dynamic Behaviours in Distributed Learning Communities”, Proceedings of 16th Australian International Education Conference, Hobart, Tasmania, Australia, (2002), September 30 – October 4.

[3] ADL, Advanced Distributed Learning, “SCORM (2004), ADL Sharable Content Object Reference Model, Version 1.3.”, http://www.adlnet.org, (2004).

[4] IMS Global Learning Consortium, Inc., “IMS Learning Design Information Model”, http://www.imsglobal.org/learningdesign/ldv1p0/imsld_infov1p0.html, (2003), January 20.

[5] IEEE LTSC, Learning Technology Standards Commitee, “Draft Guidelines for Learning Technology”, http://ltsc.ieee.org, (2001), January 30.